IOSR Journal of Electronics and Communication Engineering (IOSR-JECE)
e-ISSN: 2278-2834,p- ISSN: 2278-8735.
PP 152-157
www.iosrjournals.org

Fpga Implementation of Low Complexity Test Circuits Using Shift Registers

Mohammed Yasir, Shameer.S
(M.Tech in Applied Electronics,MG University College Of Engineering,Muttom,Kerala,India) (M.Tech in Applied Electronics,MG University College Of Engineering,Muttom,Kerala,India)

ABSTRACT: *Functional broadside tests are two-pattern scan based tests that avoid over testing by ensuring that a circuit traverses only reachable states during the functional clock cycles of a test. On-chip test generation has the advantage that it reduces test data volume and facilitates at-speed test application. This paper shows that on-chip generation of functional broadside tests can be done using a simple and fixed hardware structure. With the proposed on-chip test generation method, the circuit is used for generating reachable states during test application. Because of the high device counts and limited input/output access that characterize VLSI circuits, conventional testing approaches are often ineffective and insufficient for VLSI circuits. Built-in self-test (BIST) is a commonly used design technique that allows a circuit to test itself. BIST has gained popularity as an effective solution over circuit test cost; test quality and test reuse problems. So here we are presenting an implementation of a tester circuit using Verilog and then implementation using FPGA.*

Keywords - *Built-in self-test , Field Programmable Gate Array, Linear feedback shift register, Multiple-Input Signature Register*

I. INTRODUCTION

Very Large Scale Integration (VLSI) has made a dramatic impact on the growth of integrated circuit technology. It has not only reduced the size and the cost but also increased the complexity of the circuits. The positive improvements have resulted in significant performance/cost advantages in VLSI systems. There are, however, potential problems which may retard the effective use and FPGA implementation of Low complexity hardware test using shift registers of future VLSI technology. Among these is the problem of circuit testing, which becomes increasingly difficult as the scale of integration grows Because of the high device counts and limited input/output access that characterize VLSI circuits, conventional testing approaches are often ineffective and insufficient for VLSI circuits. It is possible now to virtually implement and check the circuit inside that computer. So the HDL has been revolutionary in the era of electronics technology. Built-in self-test (BIST) is a commonly used design technique that allows a circuit to test itself. BIST has gained popularity as an effective solution over circuit test cost; test quality and test reuse problems.

Testing of integrated circuits (ICs) is of crucial importance to ensure a high level of quality in product functionality in both commercially and privately produced products. The impact of testing affects areas of manufacturing as well as those involved in design. Given this range of design involvement, how to go about best achieving a high level of confidence in IC operation is a major concern. This desire to attain a high quality level must be tempered with the cost and time involved in this process. These two design considerations are at constant odds. It is with both goals in mind (effectiveness vs. cost/time) that Built-In-Self Test (BIST) has become a major design consideration in Design-For-Testability (DFT) methods.

Automatic test equipments (ATE) that are used for testing the circuits are relatively expensive. This hinders the use of automatic test equipments in smaller applications. Also with the increased level of integration ATEs became inefficient. Since the Automatic test equipments are testing the entire circuit there is an increase of test time. Due to the development of VLSI the amount of volume confined in a small region has also increased considerably leading to an increase of test data volume. Also many of the modern embedded SoC (system on chip) systems cannot be accessed directly from outside the chip. With increasing integration density, the amount of manufacture faults is increasing due to these issues it became necessary to develop a system that is capable to generate self testing mechanism. Thus the idea of a new DFT technique, BIST came up, with which it became possible to generate patterns for self test. With BIST it became possible to generate a uniform test mechanism for production, systems and maintenance. Also dynamic test of system properties became possible. It can be tested during operation and maintenance.

The typical BIST architecture is composed of three hardware modules in addition to the circuit under test (CUT).The Test Pattern Generator (TPG) generates the test patterns for the CUT. The Output Response Analyzer (ORA) compares or analyzes the test responses to determine correctness of the CUT. The BIST controller is the central unit to control all the BIST operations including initialization and length of the BIST sequence. In a BIST system hierarchy, often there are BIST controllers at each level of the circuit hierarchy,such as module, chip, board, and system levels. Each BIST controller is responsible for the self-test in that particular level, the control of BIST operations for the lower level BIST, and the reporting of the test results to the upper level. The design of a TPG is determined by the test strategy being deployed. The test strategy being selected is determined by the fault coverage, test hardware overhead, and testing time.

The main drivers for the widespread development of BIST techniques are the fast-rising costs of ATE testing and the growing complexity of integrated circuits. It is now common to see complex devices that have functionally diverse blocks built on different technologies inside them. Such complex devices require high-end mixed-signal testers that possess special digital and analog testing capabilities. BIST can be used to perform these special tests with additional on-chip test circuits, eliminating the need to acquire such high-end testers.BIST is also the solution to the testing of critical circuits that have no direct connections to external pins. devices. In the near future, even the most advanced tester may no longer be adequate for the fastest chip, a situation wherein self-testing may be the best solution for it. Figure shows a simple BIST block diagram which uses a linear feedback shift register (LFSR) to generate the test vectors and multiple input signature register (MISR) to verify the output against the correct response of the circuit under test.

II. BIST ARCHITECTURE

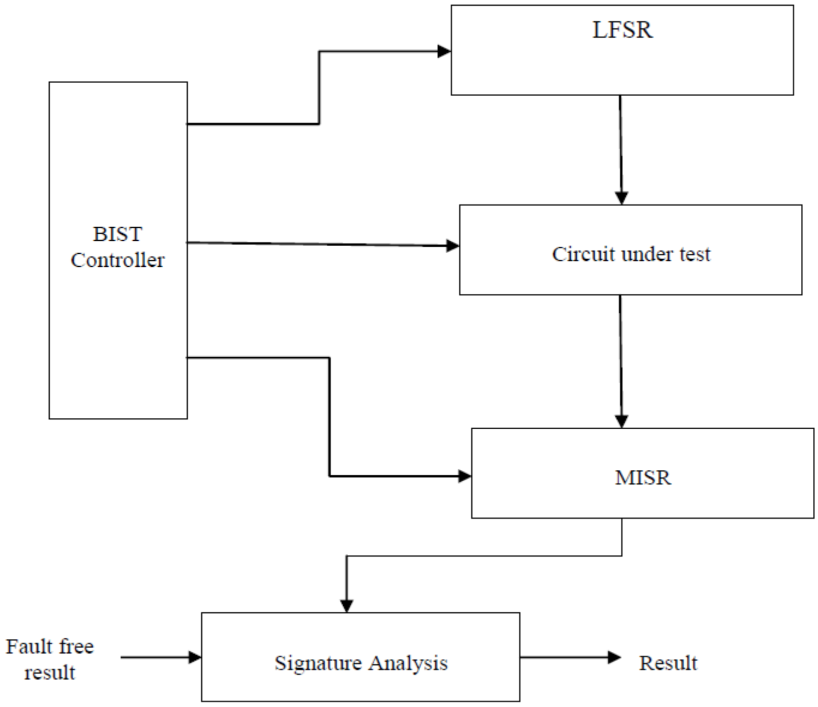


Fig 2.1: BITS architecture

2.1 Linear feedback shift register

When the LFSR starts, the LFSR is reset to zero first. Then the seed value is applied sequentially from I0. As the LFSR is operating in test pattern generation mode, the I0 is set to 0. Different structures pattern. It means that if the BIST time is limited, the structure of LFSR will affect the BIST time and Fault Coverage (FC) of Circuit under Test (CUT).

LFSR is an n-bit shift register which pseudo-randomly scrolls between 2n-1 values, but does it very quickly because there is minimal combinational logic involved. The all zeros case is not possible in this type of LFSR, but the probability of any bit being "1" or "0" is 50% except for that. Therefore, the sequence is pseudorandom in the sense that the probability of a "1" or "0" is approximately 50%, but the sequence is repeatable. Like a binary counter, all 2n - 1 states are generated, but in a “random” order that is repeatable. The exclusive-OR gates and shift register act to produce a pseudorandom binary sequence (PRBS) at each of the flip-flop outputs. By correctly choosing the points at which we take the feedback from an n -bit shift register we can produce a PRBS of length 2n – 1, a maximal-length sequence that includes all possible patterns (or vectors) of n bits, excluding the all-zeros pattern. In an LFSR, the bits contained in selected positions in the shift register are combined in some sort of function and the result is fed back into the register's input bit.

Suppose a BIST application required a certain set of test vector sequences but not all the possible 2n – 1 pattern generated using a given primitive polynomial – this is where a generic LFSR design would find application. Making use of such an implementation would make it possible to reconfigure the LFSR to implement a different primitive/non-primitive polynomial on the fly.

2.2 Circuit under Test

Here in our thesis for demonstration we used adder as the test circuit. Simulation for original or true circuit is done first, and then simulation is done for the false circuit or circuit under test. The output from LFSR is fed to the circuit under test (CUT). A sequence of output vectors are produced based on the characteristic function of the CUT. This outputs act as the input for the next section i.e. MISR. The MISR evaluates this output producing a sequence of analysis vectors. In actual practice the circuit under test will be a highly integrated VLSI circuit, and hence the evaluation purpose requires the use of an MISR with greater number of pins.

2.3 MISR – multiple-input signature register

A serial-input signature register can only be used to test logic with a single output. The idea of a serial input signature register can be extended to multiple-input signature register (MISR). There are several ways to connect the inputs of LFSRs to form an MISR..Since the XOR operation is linear and associative, (A xor B) xor C = A xor (B xor C), as long as the result of the additions are the same then the different representations are equivalent. If we have an n -bit long MISR we can accommodate up to n inputs to form the signature. If we use m < n inputs we do not need the extra XOR gates in the last n – m positions of the MISR. MISR reduce the amount of hardware required to compress a multiple bit stream. LFSR and/or MISR circuit is implemented using a memory already existing in a circuit to be tested.

The example above considered a signature analyzer that had a single input, but the same logic is applicable to a CUT that has more than one output. This is where the MISR is used. This is obtained by adding XOR gates between the inputs to the flip-flops of the SAR for each output of the CUT.

2.4 Signature analysis

Signature Analysis is a compression technique based on the concept of cyclic redundancy checking. The good and faulty circuits produce different signatures. Test Patterns for BIST can be generated at-speed by an LFSR with only a clock input. The outputs of the circuit-under-test must be compared to the known good response. In general, collecting each output response and off-loading it from the circuit under test for comparison is too inefficient to be practical. The general solution is to compress the entire output stream into a single signature value.

When test patterns are applied to a CUT, its fault free response(s) should be pre-determined. For a given set of test vectors, applied in a particular order, we can obtain the expected responses and their order by simulating the CUT. These responses may be stored on the chip using ROM, but such a scheme would require a lot of silicon area to be of practical use. Alternatively, the test patterns and their corresponding responses can be compressed and re-generated, but this is of limited value too, for general VLSI circuits due to the inadequate reduction of the huge volume of data. The solution is compaction of responses into a relatively short binary sequence called a signature.

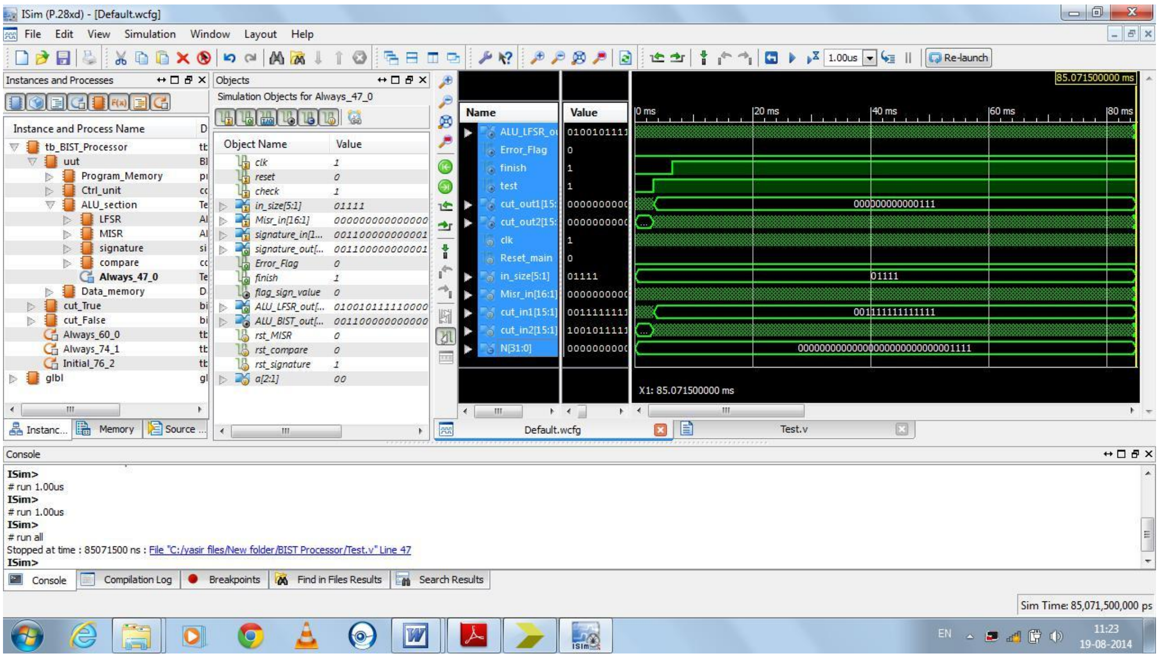
2.5 Fault detection

For example if we have three input bits and three output bits for the given circuit to be tested. So we design an LFSR with three output bits to generate the test vectors. Let the LFSR be of maximal length so that we get the test vector sequence or pseudo random binary sequence as {111, 011, 001, 100, 010, 101, 110} as explained earlier. This pseudo random binary sequence is applied to the circuit under test and the output is recorded. The output at the end of 7th clock is taken as the signature.

III. RESULTS AND DISCUSSIONS

Codings for modules like LFSR, MISR and BIST processor are finished. The simulation result for the same is shown.Also the same results were obtained using FPGA.

3.1 Simulation results of verilog implementation



Simulation Objects for Always_47_0

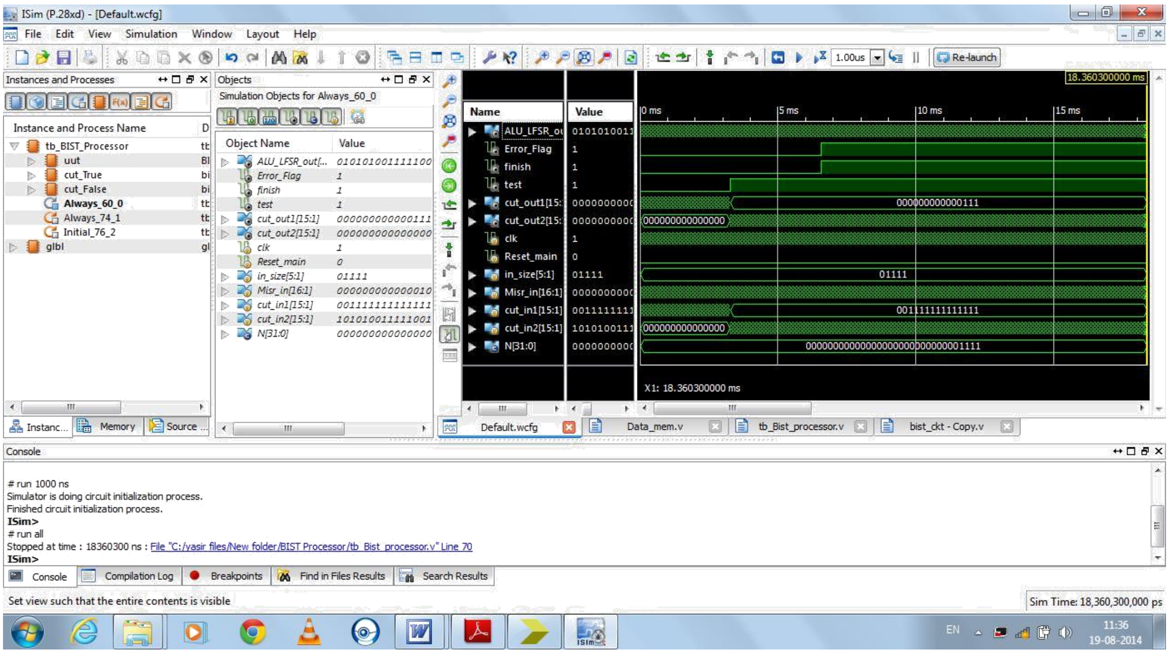
Object Name	Value
clk	1
reset	0
check	1
in_size[5:1]	01111
Misr_in[16:1]	000000000000000
signature_in[1...	001100000000001
signature_out[...	001100000000001
Error_Flag	0
finish	1
flag_sign_value	0
ALU_LFSR_out[...	010010111110000
ALU_BIST_out[...	001100000000000
rst_MISR	0
rst_compare	0
rst_signature	1
a[2:1]	00

Name	Value
ALU_LFSR_ou	0100101111
Error_Flag	0
finish	1
test	1
cut_out1[15:	0000000000
cut_out2[15:	0000000000
clk	1
Reset_main	0
in_size[5:1]	01111
Misr_in[16:1]	0000000000
cut_in1[15:1]	0011111111
cut_in2[15:1]	1001011111
N[31:0]	0000000000

```
ISim>
# run 1.00us
ISim>
# run 1.00us
ISim>
# run all
Stopped at time : 85071500 ns : File "C:/yasir files/New folder/BIST Processor/Test.v" Line 47
ISim>
```

Sim Time: 85,071,500,000 ps

Figure 3.1: Simulation results for original circuit



Simulation Objects for Always_60_0

Object Name	Value
ALU_LFSR_out[...	010101001111100
Error_Flag	1
finish	1
test	1
cut_out1[15:1]	000000000000111
cut_out2[15:1]	000000000000000
clk	1
Reset_main	0
in_size[5:1]	01111
Misr_in[16:1]	000000000000010
cut_in1[15:1]	001111111111111
cut_in2[15:1]	101010011111001
N[31:0]	000000000000000

Name	Value
ALU_LFSR_ou	0101010011
Error_Flag	1
finish	1
test	1
cut_out1[15:	0000000000
cut_out2[15:	0000000000
clk	1
Reset_main	0
in_size[5:1]	01111
Misr_in[16:1]	0000000000
cut_in1[15:1]	0011111111
cut_in2[15:1]	1010100111
N[31:0]	0000000000

```
# run 1000 ns
Simulator is doing circuit initialization process.
Finished circuit initialization process.
ISim>
# run all
Stopped at time : 18360300 ns : File "C:/yasir files/New folder/BIST Processor/tb_Bist_processor.v" Line 70
ISim>
```

Set view such that the entire contents is visible

Sim Time: 18,360,300,000 ps

Figure 3.2: Simulation for faulty circuit

IV. APPLICATIONS

BIST is commonplace in weapons, avionics, medical devices, automotive electronics, complex machinery of all types, unattended machinery of all types, and integrated circuits.

4.3.1 Automotive

Automotive tests itself to enhance safety and reliability. For example, most vehicles with antilock brakes test them once per safety interval. If the antilock brake system has a broken wire or other fault, the brake system reverts to operating as a normal brake system. Most automotive engine controllers incorporate a "limp mode" for each sensor, so that the engine will continue to operate if the sensor or its wiring fails. Another, more trivial example of a limp mode is that some cars test door switches, and automatically turn lights on using seat-belt occupancy sensors if the door switches fail.

4.3.2 Aviation

Almost all avionics now incorporate BIST. In avionics, the purpose is to isolate failing line-replaceable units, which are then removed and repaired elsewhere, usually in depots or at the manufacturer. Commercial aircraft only make money when they fly, so they use BIST to minimize the time on the ground needed for repair and to increase the level of safety of the system which contains BIST. Similar arguments apply to military aircraft. When BIST is used in flight, a fault causes the system to switch to an alternative mode or equipment that still operates. Critical flight equipment is normally duplicated, or redundant. Less critical flight equipment, such as entertainment systems, might have a "limp mode" that provides some functions.

4.3.3 Electronics

BIST is used to make faster, less-expensive integrated circuit manufacturing tests. The IC has a function that verifies all or a portion of the internal functionality of the 25 IC. In some cases, this is valuable to customers, as well. For example, a BIST mechanism is provided in advanced field bus systems to verify functionality. At a high level this can be viewed similar to the PC BIOS's power-on self-test (POST) that performs a self-test of the RAM and buses on power-up.

4.3.4 Computers

The typical personal computer tests itself at start-up (called POST) because it's a very complex piece of machinery. Since it includes a computer, a computerized self-test was an obvious, inexpensive feature. Most modern computers, including embedded systems, have self-tests of their computer, memory and software.

4.3.5 Unattended Machinery

Unattended machinery performs self-tests to discover whether it needs maintenance or repair. Typical tests are for temperature, humidity, bad communications, burglars, or a bad power supply. For example, power systems or batteries are often under stress, and can easily overheat or fail. So, they are often tested.

Often the communication test is a critical item in a remote system. One of the most common, and unsung unattended system is the humble telephone concentrator box. This contains complex electronics to accumulate telephone lines or data and route it to a central switch. Telephone concentrators test for communications continuously, by verifying the presence of periodic data patterns called frames. Frames repeat about 8,000 times per second.

4.3.6 Medicine

Medical devices test themselves to assure their continued safety. Normally there are two tests. A power-on self-test (POST) will perform a comprehensive test. Then, a periodic test will assure that the device has not become unsafe since the power-on self test. Safety-critical devices normally define a "safety interval", a period of time too short for injury to occur. The self test of the most critical functions normally is completed at least once per safety interval. The periodic test is normally a subset of the power-on self test.

4.3.7 Military

One of the first computer-controlled BIST systems was in the U.S.'s Minuteman Missile. Using an internal computer to control the testing reduced the weight of cables and connectors for testing. The Minuteman was one of the first major weapons systems to field a permanently installed computer-controlled self-test.

V. CONCLUSION AND FUTURE SCOPE

Very Large Scale Integration (VLSI) has made a dramatic impact on the growth of integrated circuit technology. It has not only reduced the size and the cost but also increased the complexity of the circuits. The positive improvements have resulted in significant performance/cost advantages in VLSI systems. There are, however, potential problems which may retard the effective use and growth of future VLSI technology. Among these is the problem of circuit testing, which becomes increasingly difficult as the scale of integration grows. Because of the high device counts and limited input/output access that characterize VLSI circuits, conventional testing approaches are often ineffective and insufficient for VLSI circuits. Built-in self-test (BIST) is a commonly used design technique that allows a circuit to test itself. BIST has gained popularity as an effective solution over circuit test cost; test quality and test reuse problems.

BIST is fast becoming an alternative solution to the rising costs of external electrical testing and increasing complexity of devices. This approach will find greater deployment in a wider variety of circumstances as more and better BIST techniques are developed. This does not mean, however, that BIST will eventually replace external electrical testing altogether. Still, BIST proponents are optimistic that BIST will someday be the preferred mode of testing, instead of being merely an alternative to external ATE testing as it is today. It can support testing in the wafer, after packaging after assembling a chip in the board, during operation and maintenance.

In this paper we have illustrated an implementation of BIST logic using Verilog. LFSR is used as a pseudorandom sequence generator. Signature analysis is used to make verification of the circuit. Signature mismatch with the reference signature means that the circuit is faulty. However, there is a small probability that the signature of a bad circuit will be the same as a good circuit. When longer sequences are used, signature analysis gives high fault coverage.

REFERENCES

[1]. I. Pomeranz and S. M. Reddy, "*Generation of functional broadside tests for transition faults*," IEEE Trans. Comput.-Aided Design Integr. Circuits Syst., vol. 25, no. 10, pp. 2207–2218, Oct. 2006.

[2]. M. Syalet al., "*Astudy of implication based pseudo functional testing*," inProc. Int. Test Conf., 2006, pp. 1–10.

[3]. H. Lee, I. Pomeranz, and S. M. Reddy, "*On complete functional broadside tests for transition faults,"* IEEE Trans. Comput.-Aided Design Integr. Circuits Syst., pp. 583–587, 2008.

[4]. Y.-C. Lin, F. Lu, and K.-T. Cheng, "*Pseudofunctional testing*," IEEE Trans. Comput.-Aided Design Integr. Circuits Syst., pp. 1535–1546, 2006

[5]. I. Pomeranz and S. M. Reddy, "*Primary input vectors to avoid in random test sequences for synchronous sequential circuits*," IEEE Trans. Comput.-Aided Design Integr. Circuits Syst., pp. 193–197, 2008.

[6]. I. Pomeranz, "*Built-in generation of functional broadside tests*," presented at the Design Autom. Test Euro. Conf., Grenoble, France

[7] Y. Zorian and A. Ivanov, "An effective BIST scheme for ROMs," *IEEE Trans. Comput.*, vol. 41, no. 5, May 1992.

[8] K. K. Saluja, R. Sharma, and C. R. Kime, "A concurrent testing technique for digital circuits," *IEEE Trans. Computer-Aided Design*, vol. 7, no. 12, p. 1259, Dec. 1988.

[9] "Concurrent Comparative Built-In Testing of Digital Circuits," Department of Electrical and Computer Engineering University of Wisconsin, Tech. Rep. ECE-8711, 1986.

[10] "Concurrent comparative testing using BIST resources," in *IEEE Int. Conf. Computer Aided Design*, Nov. 1987, pp. 336–339.

[11] I. Voyiatzis, D. Nikolos, A. Paschalis, C. Halatsis, and T. Haniotakis, "An efficient comparative concurrent built-in self test technique," in 4^{th} *IEEE Asian Test Symp.*, Bangalore, India, Nov. 1995.